REMARKS

Applicants would like to thank the Examiner for the thorough examination of the present application. The arguments supporting patentability of the claims are provided below.

I. The Claims

The present invention, as recited in independent Claim 1, for example, is directed to a method for patching read only memory (ROM) instructions in an electronic system comprising a first non-volatile memory portion storing instruction groups defining patching functionalities, an extended memory portion storing extended instructions, and an additional memory portion.

The method comprises checking a flag stored in the additional memory portion. The flag indicates a need for executing the extended instructions in the extended memory portion. Processing of the ROM instructions in the first non-volatile memory portion and the extended instructions in the extended memory portion are alternated based upon the flag.

Independent method Claim 16 is similar to independent method Claim 1. Independent device Claim 25 is similar to independent method Claim 1.

II. The Claims Are Patentable

The Examiner rejected independent Claims 1, 16 and 25 over the Wong et al. published patent application. Wong et al. discloses a programmable memory that stores patches and vectors to determine a patch address. Paragraph 7 in Wong et al. provides: "The programmable memory stores patch information including patch code and one or more patch

vectors. Each patch vector includes a break-out address from the ROM and a patch-in address to a corresponding location within the patch code".

In sharp contrast, the claimed invention determines a patch address by executing a subroutine stored in a fixed address of a non-volatile memory. The subroutine returns an address of a patch code. Instructions are predisposed for the patching according to which it is possible to call indirectly a patch (or extension) into the non-volatile memory. An advantage in avoiding the use of patch vectors is to avoid comparisons of break-out addresses with addresses provided by the processor.

Another advantage is with respect to hardware since the patch vectors according to Wong et al. require the use of a patch controller and a corresponding modification to the electronic embedded architecture. Wong et al. further provides in paragraph 7: "The patch controller is operative to compare an address provided by the processor with each break-out address to determine a breakout condition, and to control the selector to transfer the processor to a corresponding location within the patch code in response to a break-out condition." In paragraph 31, Wong et al. further provides: "A breakout condition occurs when a break-out address in the patch table **419** matches the current address on the POC bus **412**."

Moreover, the claimed invention is stronger with respect to security as compared to Wong et al. According to Wong et al., the ROM code can be patched by means of the above mentioned patch controller. This means that anyone who is in charge to write the patch is allowed to modify the content of the ROM code, even if the ROM code should be protected from

any modification (i.e., special Telecom Authentication Algorithms that cannot be modified once masked on the ROM code). In the claimed invention, the possibility to choose whether or not to make patchable a piece of ROM code is opened is the scope of the subroutine, thus preventing patching of whatever a programmer wants to patch.

Wong et al. thus teaches that the patch controller controls operation to determine "when and where to break out" from the original program code, but it does not specify how. Paragraph 32 of Wong et al. provides: "It is appreciated that any portion up to all of the original code may be superceded by the patch code **420**. The patch controller **410** controls operation of the UPC **402** to determine when and where to break out of the original program code operation from the ROM **404** and into patch code **420** and when and where to return back into the original code."

In sharp contrast, the Applicants' specification describes "when and where" break out occurs, from an original program, through the flag. Paragraph 27 in the Applicants specification provides: "In the additional read/write volatile memory portion **5**, for each application patch code is present information (called a flag) that keeps the status of its execution. The flag may assume two types of value: free or busy (green or read, respectively)".

The flag is a binary data stored in RAM, assuming two possible states: green or red (0 or 1 in binary logic). Each predisposed ROM subroutine has its corresponding "execution flag". Paragraph 34 in the Applicants specification provides "In more detail, when the ROM subroutine starts (0) with a green flag (initial status), the following actions are

taken. The ROM subroutine gets the information about the presence of the EEPROM subroutine (1). If the EEPROM subroutine is not present, then the ROM subroutine proceeds with the normal execution of ROM instructions (9) until the end of the subroutine."

Paragraphs 35 and 36 in the Applicants' specification further provide: "In the case of the EEPROM subroutine being present, the following are performed. The ROM subroutine checks the status of the flag. If the flag is red then the ROM subroutine proceeds with the normal execution of ROM instructions (9) until the end of subroutine (10) by passing, for the return of the control of the ROM subroutine (7) and finally for resetting the flag to the green status. For a green flag, the following are performed. The ROM subroutine sets the flag to a red status (3). The ROM subroutine calls the EEPROM subroutine (4). The EEPROM subroutine calls the calling ROM subroutine (0). In this case the EEPROM subroutine reuses the ROM instructions. This step could be processed even before EEPROM instructions are executed (5) or both, that is, before and after step (5). The EEPROM subroutine returns the control to ROM subroutine (7), and the ROM subroutine sets the flag to a green status (8), and the ROM subroutine ends its execution."

The essential functionality of the flags is to control the execution flow of the ROM subroutine. The ROM subroutine, if predisposed by the ROM programmer, always evaluates the content of its corresponding flag before prosecuting the execution. More particularly, if the flag is red the ROM subroutine continues its execution without considering the EEPROM content. This is a case of no

alteration of the ROM code or a case of calling by a corresponding EEPROM subroutine.

If the flag is green the ROM subroutine continues its execution by passing through the EEPROM subroutine to be called. So, the ROM sets the flag to red and calls the EEPROM subroutine. During the execution of EEPROM subroutine, the "patch code" (the content of subroutine) can or cannot call, in according to its purposes, the calling ROM subroutine. This is a case of alteration of the ROM code. In this last case, if the ROM subroutine is called by the corresponding EEPROM subroutine, the ROM subroutine continues its execution without incurring a recursive action. It finds the flag at red (set before) and continues as indicated in the case "if the flag is red". The status of flag (from red to green) is restored by ROM subroutine at the end of its execution. The reusing of the ROM code, furthermore, provides an advantage of making the EEPROM patch code smaller.

In accordance with the claimed invention, when an example is reported, the patch mechanism is compared with flags with respect to the patch mechanism based on patch vector (as disclosed in Wong et al.). A ROM subroutine computing a simple arithmetic operation is supposed to be patched, adding a new operation to the original ROM subroutine.

It is further noted that Wong et al. needs to map a portion of the patch code. Paragraph 33 of Wong et al. provides: "In this case, the patch code **420** located in the RAM **406** is outside the allowed operation instruction space of the UPC **402**. In this case, the patch code **420** is mapped into unused ROM space". In sharp contrast, the claimed invention

does not need to map any portion of patch code.

In conclusion, Wong et al. describes a low-level patch solution based on new electronic embedded architecture, and patch code that can modify the ROM code. In sharp contrast, the claimed invention provides an application level patch approach adapted for execution flow fags and patch codes that can only modify specific ROM code.

Accordingly, it is submitted that independent Claim 1 is patentable over the Wong et al. published patent application. Independent Claims 16 and 25 are similar to independent Claim 1. Therefore, it is submitted that these claims are also patentable over the Wong et al. published patent application.

In view of the patentability of independent Claims 1, 16 and 25, it is submitted that the dependent claims, which include yet further distinguishing features of the invention are also patentable. These dependent claims need no further discussion herein.

III. CONCLUSION

In view of the arguments provided herein, it is submitted that all the claims are patentable. Accordingly, a Notice of Allowance is requested in due course. Should any minor informalities need to be addressed, the Examiner is encouraged to contact the undersigned attorney at the telephone number listed below.

In Re Patent Application of:
D'ALBORE ET AL.
Serial No. **10/820,462**
Filing Date: **April 8, 2004**
_____/

Respectfully submitted,

MICHAEL W. TAYLOR
Reg. No. 43,182
Allen, Dyer, Doppelt, Milbrath
& Gilchrist, P.A.
255 S. Orange Avenue, Suite 1401
Post Office Box 3791
Orlando, Florida 32802
407-841-2330